PUBLIC COMPLAINT SORTING

A Mini Project Report

submitted by

ASWATHY V (MES20MCA-2012)

to

the APJ Abdul Kalam Technological University
in partial fulfillment of the requirements for the award of the Degree

of

Master of Computer Applications

Department of Computer Applications

MES College of Engineering
Kuttippuram, Malappuram - 679 582

February 2022

DECLARATION

I undersigned hereby declare that the project report **PUBLIC COMPLAINT SORTING**, submitted for partial fulfillment of the requirements for the award of degree of Master of Computer Applications of the APJ Abdul Kalam Technological University, Kerala, is a bona fide work done by me under supervision of **Mr.BALACHANDRAN K P**, Assistant Professor, Department of Computer Applications. This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

Place:KUTTIPURAM

Date:28-02-2022

ASWATHY V (MES20MCA-2012)

DEPARTMENT OF COMPUTER APPLICATIONS
MES COLLEGE OF ENGINEERING, KUTTIPPURAM

CERTIFICATE

This is to certify that the report entitled **PUBLIC COMPLAINT SORTING** is a bona fide record of the Mini Project work carried out by **ASWATHY V (MES20MCA-2012)** submitted to the APJ Abdul Kalam Technological University, in partial fulfillment of the requirements for the award of the Master of Computer Applications, under my guidance and supervision. This report in any form has not been submitted to any other University or Institution for any purpose.

Internal Supervisor(s) External Supervisor(s)

Head Of The Department

Acknowledgements

My endeavor stands incomplete without dedicating my gratitude to a few people who have contributed towards the successful completion of our project.I pay my gratitude to the Almighty for his invisible help and blessing for the fulfillment of this work.At the outset i express my heartful thanks to **Mr.Balachandran KP** for his valuable guidance and supervision. I take this opportunity to express my profound gratitude to **Mrs.Priya JD**, my group tutor as well as my project coordinator for her valuable support, timely advise and strict schedules to complete my project. I also grateful to all my teaching and non-teaching staff for their encouragement, guidance and whole-hearted support. Last but not least, We gratefully indebted to my family and friends, who give me a precious help in doing my project

ASWATHY V (MES20MCA-2012)

Abstract

Reporting any Civic or day to day life problems has no longer been an easy process for the citizens. They have to follow a long procedure and formalities to register their problems or to report such problems like street damages, garbage management problems (garbage bin over owing), Electricity problem, Water problem etc. in short citizens can post their problems which come under the surveillance of municipal. There is still no guarantee that the reported grievances would be resolved or addressed by the concerned municipal department authority. That is why most of the time complaints go unheard, unanswered and unresolved usually because peoples are very busy with their day to day work and they don't have time to report the complaints and to follow the time consuming process, citizens are not taking initiative to register problems. To make an easy reporting system for complaining procedure ,we are going to implement an machine learning online web application that will provide a platform for citizens to rise their voice against civic issues and report their problems with infrastructure in their city to relevant municipal department, So whenever people come across any civic issue in city infrastructure or any daily life disturbance they can share ,discuss and get resolved the problems by concerned departments authority by means of this online web portal. Citizens can share their ideas, suggestions with each other and they can also view the problems posted by other citizens.

Keywords:

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

improve Infrastructure and condition of our city and to make people take initiative to rise their voice against civic issues which they face into their daily life we are developing this system. Which will help to build a unity or strong bond within citizens. System is providing platform for citizens where they can report problems, share ideas and suggestions. It will be helpful to collect valuable source as feedback from citizens about progress improvement of city through the different posts or images posted by citizens. This is flexible and interactive interface for people to use application for register complaints and to share ideas, this system to provide flexible communication platform for citizens.

1.1.1 Motivation

1.2 Objective

This project is used to resolve problem in less time and to keep track on all process which will going on after registering particular complaints. Municipal authorities and users both will get notifications from each other. To develop this system, we use machine learning and image processing. System is providing platform for citizens where they can report problems, share ideas and suggestions. It will be helpful to collect valuable source as feedback from citizens about progress improvement of city through the different posts or images posted by citizens.

It will be helpful to resolve posted problems in limited time. The main purpose of the system to resolve problem in less time and to keep track on all process which will going on after registering particular complaints. Municipal authorities and users both will get notifications from each other.

1.3 Report Organization

The project report is divided into four sections. Section 2 describes the existing system that is the current scenario. Section 3 describes the methodology used for implementing the project. In methodology, workflow of the project, and sprints details are described. Finally Section 4 gives the conclusion.

Chapter 2

Literature Survey

Towards Two-Tier Citizen Sensing, Citizen Sensing is a powerful paradigm involving citizens collectively participating in data collection. The pervasiveness of mobile devices has taken citizen sensing to unprecedented levels of adoption, as anyone with a phone can easily participate. A delegated authorization solution for smart-city mobile, An increasingly popular scenario for Smart Cities is the one in which mobile apps attempt to access resources (e.g., open data about public transportation or egovernment services) made available by city authorities through the use of Application Programming Interfaces (APIs). There is a growing awareness of the benefits of using APIs to foster civic engagement through a more efficient and personalized delivery of government services, and as an enabler of a new wave of innovation contributing to a more automated and sustainable city functioning. Tools enabling online contributions by older adults In this paper they implemented tool for contribution from older adults citizens.The results are of particular importance for the development of technology that aims at reducing social isolation for people with less chances to interact, such as older adults. Service-learning project for computing students: Creating a mobile app for a non-profit agency, Over the last 15 years the world as seen a surge in users owning smartphones and electronic devices. People today have smartphones that have the computational capabilities of computers from more than 45 years ago. With the growing populous under this information age, individuals have access to so much data. It is a fantastic time to spread ideas as anyone can reach anybody else over the internet almost instantaneously. It is not too di cult to get into contact with anyone even if they are on the other side of the globe. This is the power of the 21st century and which has generated a team-oriented society. World Social Welfare Circumstances: Social Welfare

and Elderly Care System in the World, and Civic Technology Chisako Yamashita This paper illustrates social welfare and elderly care system and issues in three countries, Japan, Norway, and the United States, and discuss how to solve social problems by using civic technologies. Citizen emotion analysis in Smart City Applications in Smart City context are improving the quality of life of citizens through several technological interactions. These interactions can be also used to relate the citizen's emotions to city areas. Thus, the main objective of this work is to present a smart phone application. Social Media Based App Organizing Daily Events Since the primary attraction for IT developers is to build applications by reusing the existing resources, especially using mobile platforms as it is changing the way software applications are developed and accessed, the platform presented in this paper aims to keep users up to date for all of their daily events. The idea was to develop a new contemporary application for the mobile platform that will be able to integrate several social media APIs. While selecting sources and notification time, the proposed implemented platform will be able to generate a to do list of the daily events, offering high flexibility and portability. Gram Sandesh Transmission-A Web Based Information System for Farmers in This the experimentation done in order to flourish a low price and impressionable information system to provide useful information to farmers in a timely manner so as to assist their decision making process. The primary reason behind development of this system was to automate the flow of information to farmers since agriculture is the backbone of our country. Gram Sandesh Transmission is a web disciplined system which targets all sort of audience by means of its ios application (for iphone users), android application(for android users), messaging server(for basic mobile handset) and gsm based. PROBLEM D

Chapter 3

Methodology

3.1 Introduction

To improve Infrastructure and condition of our city and to make people take initiative to rise their voice against civic issues which they face into their daily life we are developing this system. Which will help to build a unity or strong bond within citizens. System is providing platform for citizens where they can report problems, share ideas and suggestions. It will be helpful to collect valuable source as feedback from citizens about progress improvement of city through the different posts or images posted by citizens. This is flexible and interactive interface for people to use application for register complaints and to share ideas, this system to provide flexible communication platform for citizens. It will be helpful to resolve posted problems in limited time. The main purpose of the system to resolve problem in less time and to keep track on all process which will going on after registering particular complaints. Municipal authorities and users both will get notifications from each other. System is using the hierarchy of different level of authority like user level then departments and their authorities and finally higher authority, this will be more effective to keep the track on each and every work related to civic issues posted by citizens. To develop this system we use machine leaning and image processing.

3.2 Modules

Module 1

• Admin can view users.

•They can add and manage department.

•They can add and manage officers.

•They can view feedback given by the users.

•They can view department wise complaints given by the users but they can't reply to the complaints.

•Taking actions against officers those who don't respond to the complaints.

Module 2

• Officer

•The officer will be added directly by the admin.

•The officer who can view the complaints given by the users and can reply for the complaints.

•Officer can change their password also.

Module 3

Users

•Users can register by themselves.

•They can send complaints and view the respective department

3.3 Developing Environment

SOFTWARE REQUIREMENTS

•Operating System : WINDOWS 10

•Front end : HTML,CSS,JAVASCRIPT

•IDE Used : Jetbrains Pycharm , Android Studio

•Technology Used : Python Java

•Frame work used : Flask

3.4 Work Flow

A Convolutional Neural Network , also known as CNN is a powerful algorithm for image processing. These algorithms are currently the best algorithms we have for the automated processing of images. Many companies use these algorithms to do things like identifying the objects in an image. There are three types of layers in CNN algorithm : 1.Convolutional Layer : In a typical neural network each input neuron is connected to the next hidden layer. In CNN, only a small region of the input layer neurons connect to the neuron hidden layer. 2.Pooling Layer : The pooling layer is used to reduce the dimensionality of the feature map. There will be multiple activation and pooling layers inside the hidden layer of the CNN. 3. Fully-Connected Layer : Fully connected layer form the last few layers in the network. The input to the fully connected layer is the output from the final pooling or Convolutional layer, which is flattened and then fed into the fully connected layer.

3.5 User Story

The project was developed using Agile Methodology.The project has two users.First one is admin and second is officer and third is user.The user story of system is given in 3:1.

User Story ID	As a <type of user>	I want to	So that I can
1	Admin	Login	Login successful with correct username and password
2	Admin	Add and Manage Officers	Add new officers and can Update and Delete all the officer details
3	Admin	View Feedback	View all the user feedback
4	User	Upload image and view result	Upload image and view result
5	User	Send feedback	Send feedback to admin
6	User	Registration	Register by themselves
7	Officer	View complaints	View complaint and replay for the complaint
8	Admin	Add and manage Department	Add new departments

Figure 3.1: userstory

3.6 project plan

User Story ID	Task Name	Start Date	End Date	Days	Status
1	Sprint 1	27-12-21	27-12-21		Completed
2		28-12-21	28-12-21	4	Completed
3	Sprint 2	29-12-21	29-12-21		Completed
4		15-01-22	16-01-22	4	Completed
5	Sprint 3	23-01-22	27-01-22		Completed
6		30-01-22	05-02-22	8	Completed
7	Sprint 4	06-02-22	10-02-22		Completed

Figure 3.2: project plan

3.7 product backlog

User Story ID	Priority <High/Medium/Low>	Size (Hours)	Sprint <#>	Status <Planned/In/Progress/Completed>	Release Date	Release Goal
1	Medium	2	1	Completed		Table Design
2	High	3		Completed		Form Design
3	High	5		Completed		Basic Coding
3	High	5	2	Completed		Creation of data set
4	Medium	5		Completed		Pre processing
5	High	5	3	Completed		Prediction
6	Medium	5		Completed		Complaint allocation Testing data
7	Medium	5	4	Completed		Testing data
8	High	5		Completed		Output generation

Figure 3.3: product backlog

3.8 sprint backlog

Backlog Item	Status & Completion date	Original estimation in Hours	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10	Day 11	Day 12
User Story #1,#2, #3			Hrs	Hrs	Hrs	Hrs	Hrs	Hrs	Hrs	Hrs	Hrs	Hrs	Hrs	Hrs
Table Design	28/12/21	2	1	1	0	0	0	0	0	0	0	0	0	0
Form Design	31/12/21	3	0	0	1	1	1	0	0	0	0	0	0	0
Basic Coding	8/1/22	5	0	0	0	0	0	1	1	1	1	1	0	0
User Story #4,#5														
Creation of Dataset	16/1/22	5	1	1	0	1	1	1	0	0	0	0	0	0
Pre processing	22/1/22	5	0	0	0	0	0	0	0	1	1	0	1	1
User Story #6,#7														
Prediction	27/1/22	5	1	1	1	0	1	1	0	0	0	0	0	0
Complaint allocation Testing data	5/2/22	5	0	0	0	0	0	0	0	1	1	1	1	1
User Story #8,#9														
Testing Data	10/2/22	5	1	1	1	1	1	0	0	0	0	0	0	0
Output Generation	20/2/22	5	0	0	0	0	0	0	2	2	1	0	0	0

Figure 3.4: sprint backlog

3.9 sprint actual

Backlog Item	Status & Completion Date	Original Estimation Hours	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10	Day 11	Day 12
User Story #1,#2, #3														
Table Design	28/12/21	2	1	1	0	0	0	0	0	0	0	0	0	0
Form Design	31/12/21	4	0	0	0	1	1	0	1	1	0	0	0	0
Coding	8/1/22	4	0	0	0	0	0	0	0	0	2	1	1	0
User Story #4,#5														
Creation of Dataset	22/01/22	5	1	1	0	1	0	1	0	0	0	0	0	0
Pre processing	27/01/22	5	1	1	1	1	1	0	0	0	0	0	0	0
User Story #6,#7														
Prediction	05/02/22	5	0	0	0	0	0	0	1	0	1	1	0	1
Complaint allocation testing data	10/02/22	5	1	1	1	1	1	0	0	0	0	0	0	0
User Story #8,#9														

Figure 3.5: sprint actual

Chapter 4

Results and Discussions

4.1 Results

This system focuses on flexible communication between citizen to citizen and citizen to respective authority. An implementation of web application in which there will be the flexible communication so that each and every citizen can raise their voice against various civic issues with the least manual interference. This application gives one to many and many to many communication bond between people. Through this website citizens can register their civic complaints in very flexible way within less time. All the issue which is been register to the web portal will be resolved within date and timing. In Fig Firstly if citizen wants to complaint regarding civic issue then he/she has to login to their account and then he/she can register the complaints but if particular citizen is new then they has to register first with some personal details. After registration he/she can login to the web portal by Aadhar card no which is unique identification and password. When any citizen posts complaints regarding any civic issue than that complaint goes to the particular department using machine learning and image processing then higher authority of that particular department can view all the complaint. When citizen register the particular issue at the same time system will generate one date behalf of user that date will nothing but the difference of 10 days from date of register issue. It will set as deadline for department authority to solve the issue in given time. As the complaints register all the citizens can give votes to it and complaints which are having highest priority will be resolved first and the complaint which is having lowest priority will be resolve but it will take some time to resolve. If in the case complaint is not resolved within the date and timing given by citizen as

well as given by the higher authority of the particular department then such a complaints will be displayed publically and these complaints will go automatically to the main authority and further action will be taken by main authority. So ultimately by this unity between the citizens will be increase the major to minor civic issue will be resolved within the time and each and every citizen can raise their voice against the civic issue with the least manual interference and within less time.

Chapter 5

Conclusions

In this project, an effective implementation for Image Processing and Machine Learning concept is used for solving Citizens problem. This project presents a conceptual architecture for a versatile, flexible and cost efficient for monitoring the citizens issues. We propose one application using machine learning and image processing in which citizens can register or post their civic issues online and they can also have assure that their problem will be resolved within the timing given by them or by authority.

References

[1] **Julio Borges, Matthias Budde, Oleg Peters, Till Riedel and Michael BeigL**, "Towards Two Tier Citizen Sensing", IEEE International Smart Cities Conference (ISC2) Year: 2016..

[2] **Giada Sciarretta, Roberto Carbone, Silvio Ranise,** "A delegated authorization solution for smart city mobile applications", IEEE 2nd International Forum on Research and Technologies for Society and Industry Leveraging a better tom orrow (RTSI) Year: 2016.

[3] **F. Ibarra, O. Korovina, M. Baez, G. Barysheva, M. March-ese, L. Cernuzzi, and F. Casati,** "Tools enabling online contributions by older adults", IEEE Internet Computing, vol. PP, no. 99, pp. 1 1, 2016..

[4] **Sonal Asija, Jean F. Coppola, ".**]] Service learning project for computing students: Creating a mobile app for a nonprofit agency", IEEE Long Island Systems, Applications and Technology Conference LISAT), 2016 "Service learning project for computing students: Creating a mobile app for a nonprofit agency", IEEE Long Island Systems, Applications and Technology Conference LISAT), 2016.

Appendix

Source Code

```
  from flask import *
import pymysql
app = Flask(__name__)
con=pymysql.connect(host='localhost',port=3306,user='root',password='',db='public complaint')
cmd=con.cursor()
app.secret_key="qwer"
@app.route('/')
def sign():
   return render_template('login.html')
@app.route('/adminhome')
def adminhome():
   return render_template('adminhome.html')
@app.route('/login',methods=['get','post'])
def login():
   uname=request.form['textfield']
   password=request.form['textfield2']
   cmd.execute("select* from login where username='"+uname+"' and password='"+password+"'")
   s=cmd.fetchone()
   if s is None:
      return '''<script>alert("invalid username or password");window.location='/' </script>'''
   elif s[3]=='admin':
      session['lid'] = s[0]
      return '''<script>alert("login successfuly");window.location='/adminhome' </script>'''
   elif s[3]=='officer':
      session['lid']=s[0]
      return '''<script>alert("login successfully");window.location='/officerhome'</script>'''
   else:
      return '''<script>alert("invalid");window.location='/' </script>'''
@app.route('/viewusers')
def viewuser():
   if 'lid' in session:
      cmd.execute("select * from user")
      s=cmd.fetchall()
      return render_template('view user table.html',val=s)
   else:
      return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/edituser',methods=['get','post'])
def edituser():
     id = request.args.get('uid')
     session['id'] = id
     cmd.execute("select * from user where user.lid='" + str(id) + "'")
     s = cmd.fetchone()
     return render_template('updateuser.html',val=s)
@app.route('/updateuser',methods=['get','post'])
```

Appendix

```
def updateuser():
   if 'lid' in session:
      fname = request.form['textfield']
      lname = request.form['textfield9']
      dob = request.form['textfield2']
      gender = request.form['radiobutton']
      place = request.form['textfield3']
      post = request.form['textfield4']
      pin =request.form['textfield5']
      phone = request.form['textfield6']
      email = request.form['textfield7']
      Aadharno = request.form['textfield8']
      cmd.execute("update user set fname='" + fname + "',lname='" + lname + "',dob='" + dob + "',gender='" + gender +
            "',post='" + post + "',pin='"+pin+"',place='" + place + "',phone='" + phone + "',email='" + email + "',`aadhar
            no`='" + Aadharno + "' where user.lid='" + str(session['id']) + "'")
      con.commit()
      return '''<script>alert("updated successfuly");window.location='/viewusers'</script>'''
   else:
      return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/edit')
def edit():
   if 'lid' is session:
      return render_template('edit.html')
   else:
      return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/deleteuser',methods=['get','post'])
def deleteuser():
   if 'lid' in session:
      id = request.args.get('uid')
      cmd.execute("delete from login where lid='" + str(id) + "'")
      cmd.execute("delete from user where lid='" + str(id) + "'")
      con.commit()
      return '''<script>alert("deleted successfuly");window.location='/viewusers' </script>'''
   else:
      return '''<script>alert("please login");window.location='/' </script>'''

@app.route('/add_managedprtmnt')
def addmanage():
   if 'lid' in session:
      cmd.execute("select * from department")
      s = cmd.fetchall()
      return render_template('department.html',val=s)
   else:
      return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/editoff',methods=['get','post'])
def editoff():
   if 'lid' in session:
      id=request.args.get('uid')
      session['id']=id
      # cmd.execute("select username from login where lid='"++"'")
      cmd.execute("select * from officers where officers.lid='"+str(id)+"'")
      s=cmd.fetchone()
      return render_template('editoff.html',val=s)
   else:
      return '''<script>alert("please login");window.location='/' </script>'''

@app.route('/updateoff',methods=['get','post'])
def updateoff():
   if 'lid' in session:
      fname = request.form['textfield']
      lname = request.form['textfield4']
      dob = request.form['textfield5']
      gender = request.form['radiobutton']
```

Appendix

```
      place = request.form['textfield6']
      post = request.form['textfield7']
      phone = request.form['textfield8']
      email = request.form['textfield9']
      Aadharno = request.form['textfield10']
      cmd.execute("update officers set
           fname='"+fname+"',lname='"+lname+"',dob='"+dob+"',gender='"+gender+"',position='"+post+"',place='"+place+"',phone='"+phone+"',email=
           where officers.lid='"+str(session['id'])+"'")
      con.commit()
      return '''<script>alert("updated successfuly");window.location='/officersview' </script>'''
 else:
       return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/deleteoff')
def deleteoff():
   if 'lid' in session:
      id=request.args.get('uid')
      cmd.execute("delete from officers where lid='"+str(id)+"'")
      cmd.execute("delete from login where lid='"+str(id)+"'")
      con.commit()
      return'''<script>alert("deleted successfuly");window.location='/officersview' </script>'''
   else:
       return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/officersview')
def officerview():
   if 'lid' in session:
      cmd.execute("SELECT officers.*,`department`.`name` FROM `department` JOIN `officers` ON
           `officers`.`departmentid`=`department`.`dpid`")
      s=cmd.fetchall()
      return render_template('officers.html',val=s)
   else:
      return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/add_manage_officer',methods=['get','post'])
def add_manage_officer():
      cmd.execute("select * from department")
      s=cmd.fetchall()
      return render_template('registration.html',val1=s)

@app.route('/regoffcr',methods=['get','post'])
def regoffcr():
   if 'lid' in session:
      try:
         depid=request.form['select']
         fname=request.form['textfield']
         lname=request.form['textfield4']
         dob=request.form['textfield5']
         gender= request.form['radiobutton']
         place= request.form['textfield6']
         post = request.form['textfield7']

         phone = request.form['textfield8']
         email = request.form['textfield9']
         Aadharno= request.form['textfield10']
         username = request.form['textfield11']
         password=request.form['textfield3']
         cmd.execute("insert into login values(null,'" + username + "','"+password+"','officer')")
         lid=con.insert_id()
    cmd.execute("insert into officers values(null,'" + str(lid)+
          "','"+fname+"','"+lname+"','"+dob+"','"+gender+"','"+post+"','"+place+"','"+phone+"','"+email+"','"+Aadharno+"','"+str(depid)+"')")
         con.commit()

         return '''<script>alert("registration successfully");window.location='/officersview'</script>'''
      except Exception as e:
         return '''<script>alert("already exist");window.location='/add_manage_officer' </script>'''
```

Appendix

```
      else:
        return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/view_feedback')
def viewfeedback():
   if 'lid' in session:
      cmd.execute("select feedback.*,user.fname,user.lname from user join feedback on feedback.lid=user.lid")
      s=cmd.fetchall()
      return render_template('userfeedback.html',val=s)
   else:
      return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/dept_complaintview',methods=['get','post'])
def complaintview():
   if 'lid' in session:
      cmd.execute("select * from department")
      a=cmd.fetchall()

      return render_template('deptsearch.html',val1=a)
   else:
      return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/searchdep',methods=['get','post'])
def searchdep():
   if 'lid' in session:
      id=request.form['select']
      con = pymysql.connect(host='localhost', port=3306, user='root', password='', db='public complaint')
      cmd = con.cursor()
      cmd.execute("select * from department")
      a = cmd.fetchall()
      cmd.execute("select complaint.*,user.fname,user.lname from user join complaint on complaint.lid=user.lid where
            `complaint`.`dp id`='"+str(id)+"'")
      s = cmd.fetchall()
     return render_template('deptsearch.html',val=s,val1=a)

   else:
      return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/view_table')
def view_table():
   if 'lid' in session:
    return render_template('view user table.html')
   else:
      return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/department')
def department():
   if 'lid' in session:
     return render_template('department.html')
   else:
      return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/add_department',methods=['get','post'])
def add_department():
   if 'lid' in session:
     return render_template('add department.html')
   else:
      return '''<script>alert("please login");window.location='/' </script>'''


@app.route('/departmentadd',methods=['get','post'])
def departmentadd():
   if 'lid' in session:
      department=request.form['textfield']
      cmd.execute("insert into department values(null,'"+department+"')")
      con.commit()
      return '''<script>alert("department added succesfully");window.location='/add_managedprtmnt'</script>'''
   else:
      return '''<script>alert("please login");window.location='/' </script>'''
```

Appendix

```
@app.route('/officers')
def officers():
   if 'lid' in session:
    return render_template('officers.html')
   else:
      return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/registration')
def registration():
   if 'lid' in session:
    return render_template('registration.html')
   else:
      return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/userfeedback')
def userfeedback():
   if 'lid' in session:
    return render_template('userfeedback.html')
   else:
      return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/deptsearch')
def deptsearch():
   if 'lid' in session:
    return render_template('deptsearch.html')
   else:
      return '''<script>alert("please login");window.location='/' </script>'''

@app.route('/sendnoti')
def sendnoti():
   if 'lid' in session:
    return render_template('sendnotification.html')
   else:
      return '''<script>alert("please login");window.location='/' </script>'''

@app.route('/sendnoti2',methods=['post','get'])
def sendnoti2():
   if 'lid' in session:
      noti=request.form['textfield']
      cmd.execute("insert into notification values(null,'"+noti+"',curdate())")
      con.commit()
      return '''<script>alert("success");window.location='/sendnoti' </script>'''
   else:
      return '''<script>alert("please login");window.location='/' </script>'''


##############################officer

@app.route('/officerhome')
def officerhome():
   if 'lid' in session:
    return render_template('officerhome.html')
   else:
      return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/viewcomplaint')
def viewcomplaint():
   if 'lid' in session:

      lid=session['lid']
      con = pymysql.connect(host='localhost', port=3306, user='root', password='', db='public complaint')
      cmd = con.cursor()
      print("SELECT `user`.`fname`,`user`.`lname`,`complaint`.*,`officers`.`departmentid` FROM `officers` INNER JOIN
            `complaint` ON `complaint`.`dp id`=`officers`.`departmentid` INNER JOIN `user` ON `user`.`lid`=`complaint`.`lid`
            WHERE `officers`.`lid`='"+str(lid)+"' AND `complaint`.`reply`='pending'")
```

Appendix

```
      cmd.execute("SELECT `user`.`fname`,`user`.`lname`,`complaint`.*,`officers`.`departmentid` FROM `officers` INNER JOIN
            `complaint` ON `complaint`.`dp id`=`officers`.`departmentid` INNER JOIN `user` ON `user`.`lid`=`complaint`.`lid`
            WHERE `officers`.`lid`='"+str(lid)+"' AND `complaint`.`reply`='pending'")
      s=cmd.fetchall()
      print(s)
      return render_template('viewcomplaint.html',val=s)
   else:
      return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/offcreply',methods=['get','post'])
def offcreply():
   if 'lid' in session:
      cid=request.args.get('id')
      session['comid']=cid
      return render_template('offcreply.html')
   else:
      return '''<script>alert("please login");window.location='/' </script>'''

@app.route('/sendreply',methods=['get','post'])
def sendreply():
   if 'lid' in session:
      id=session['comid']
      reply=request.form['textfield']
      cmd.execute("update complaint set reply='"+reply+"' where c_id='"+str(id)+"'")
      con.commit()
      return '''<script>alert("replied successfuly");window.location='/officerhome'</script>'''
   else:
      return '''<script>alert("please login");window.location='/' </script>'''

@app.route('/change_password')
def change_password():
   if 'lid' in session:
    return render_template('change password.html')
   else:
      return '''<script>alert("please login");window.location='/' </script>'''
@app.route('/passwordchange',methods=['get','post'])
def passwordchange():
   if 'lid' in session:
         offid=session['lid']

         currentpassword=request.form['textfield']

         newpassword = request.form['textfield2']

         confirmpassword = request.form['textfield3']

         cmd.execute("SELECT * FROM `login` WHERE `login`.`lid`='"+str(offid)+"' and password='"+currentpassword+"'")
         s=cmd.fetchone()

         if s is None:
           return '''<script>alert("password mismatch");window.location='/officerhome'</script>'''
         elif newpassword==confirmpassword:
           cmd.execute("update login set password='"+newpassword+"' where lid='"+str(offid)+"'")
           con.commit()
           return '''<script>alert("success");window.location='/'</script>'''
         else:
           return '''<script>alert("Changing failed");window.location='/officerhome'</script>'''
   else:
     return '''<script>alert("please login");window.location='/' </script>'''


@app.route('/viewnotification')
def viewnotification():
   cmd.execute("select * from notification")
```

Appendix

```
    s=cmd.fetchall()
    return render_template('notification.html',val=s)


@app.route('/logout')
def logout():
    session.clear()
    return render_template('login.html')
@app.route('/deletedept')
def deletedept():
    id=request.args.get('id')
    cmd.execute("delete from department where dpid='"+id+"'")
    con.commit()
    return '''<script>
    alert("deletedddd");window.location="/add_managedprtmnt"
    </script>'''
# @app.route('/deleteoffice')
# def deleteoffice():
app.run(debug=True
```

Database Design

Attribute Name	Datatype	length	Description
id	Integer	11	Primary Key
complaint	Varchar	50	Unique
date	date		
replay	varchar	50	
lid	Integer	10	

Table A.1: Complaint

Attribute Name	Datatype	length	Description
id	Integer	11	Primary Key
feedback	Varchar	50	Unique
date	date		
uid	Integer	11	

Table A.2: feedback

Attribute Name	Datatype	length	Description
id	Integer	11	Primary Key
username	Varchar	50	Unique
password	varchar	50	
type	varchar	20	

Table A.3: login

Attribute Name	Datatype	length	Description
id	Integer	11	Primary Key
notification	Varchar	50	Unique
date	date		

Table A.4: notification

Attribute Name	Datatype	length	Description
id	Integer	11	Primary Key
lid	Integer	50	Unique
fname	varchar		
lname	varchar	11	
dob	date	50	
gender	varchar		
position	varchar		
place	varchar	11	
phone	bigint	50	
email	varchar		
Adhaar number	Integer	11	

Table A.5: officers

Attribute Name	Datatype	length	Description
id	Integer	11	Primary Key
lid	Integer	50	Unique
fname	varchar		
lname	varchar	11	
dob	date	50	
gender	varchar		
position	varchar		
place	varchar	11	
phone	bigint	50	
email	varchar		
Adhaar number	Integer	11	

Table A.6: officers

DaTaflow Diagram

Attribute Name	Datatype	length	Description
id	Integer	11	Primary Key
username	varchar	50	Unique
place	varchar	50	
post	varchar	30	
pin	Integer	11	
phone no	Integer	10	
email	varchar	30	
lid	Integer	11	

Table A.7: user

Attribute Name	Datatype	length	Description
id	Integer	11	Primary Key
officerid	Integer	11	Unique
works	varchar	40	
date	date		
status	varchar	40	

Table A.8: work

5.1 User Interface

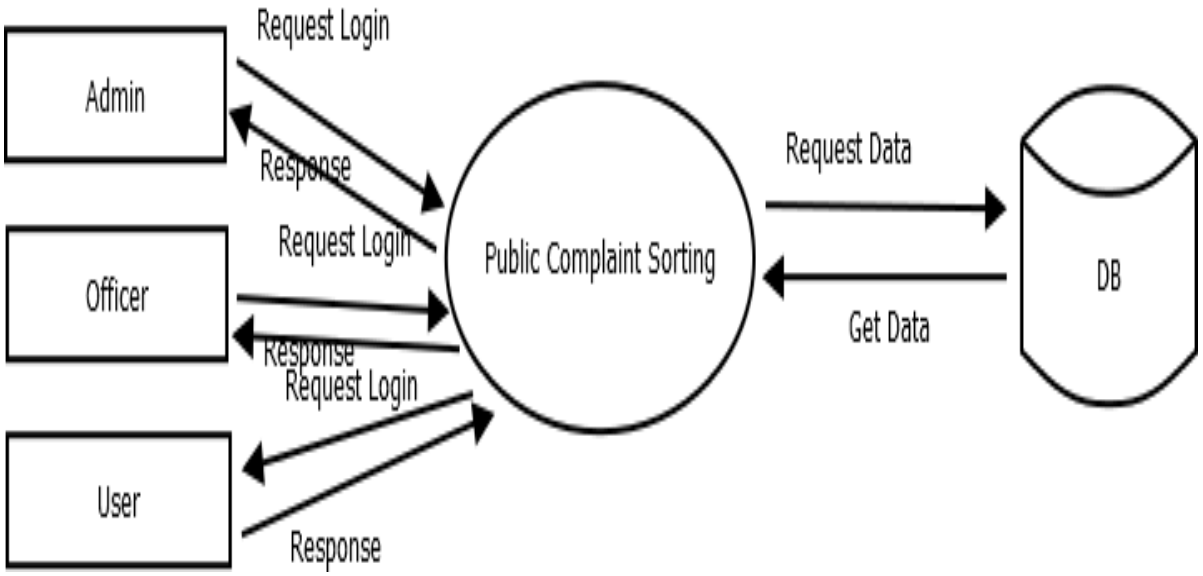

Figure A.1: Dataflow Diagram

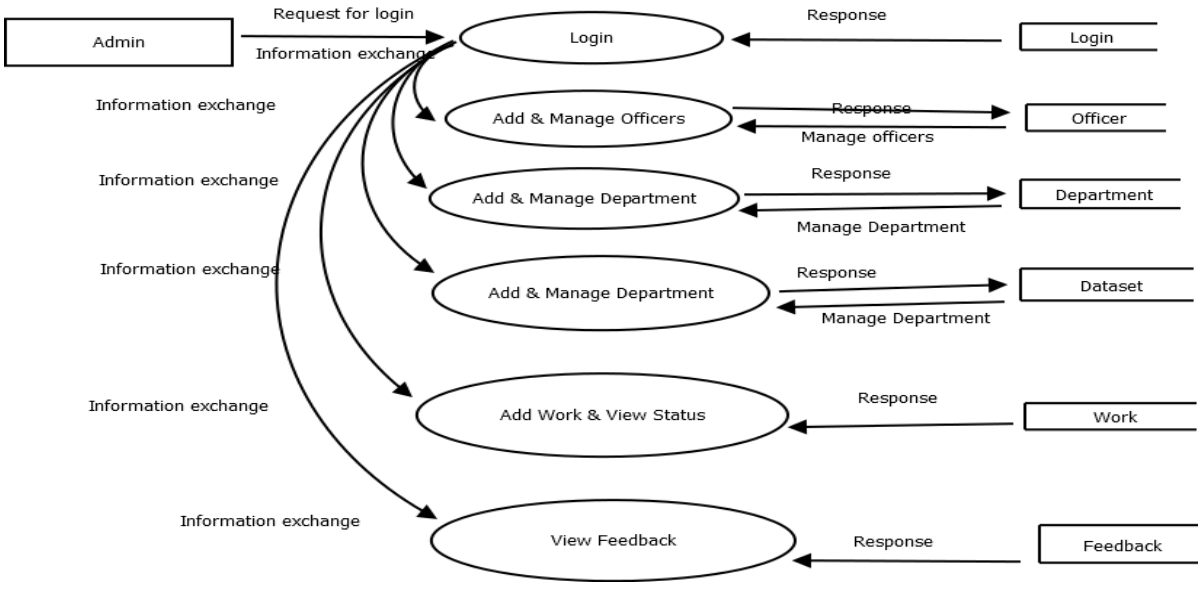

Figure A.2: Dataflow Diagram

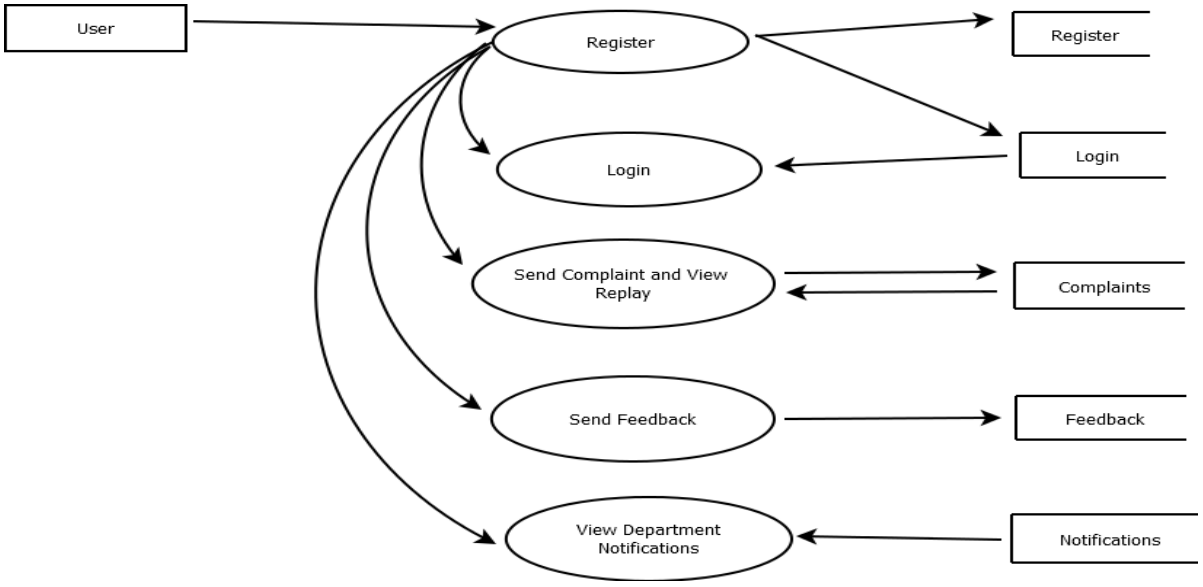

Figure A.3: Dataflow Diagram

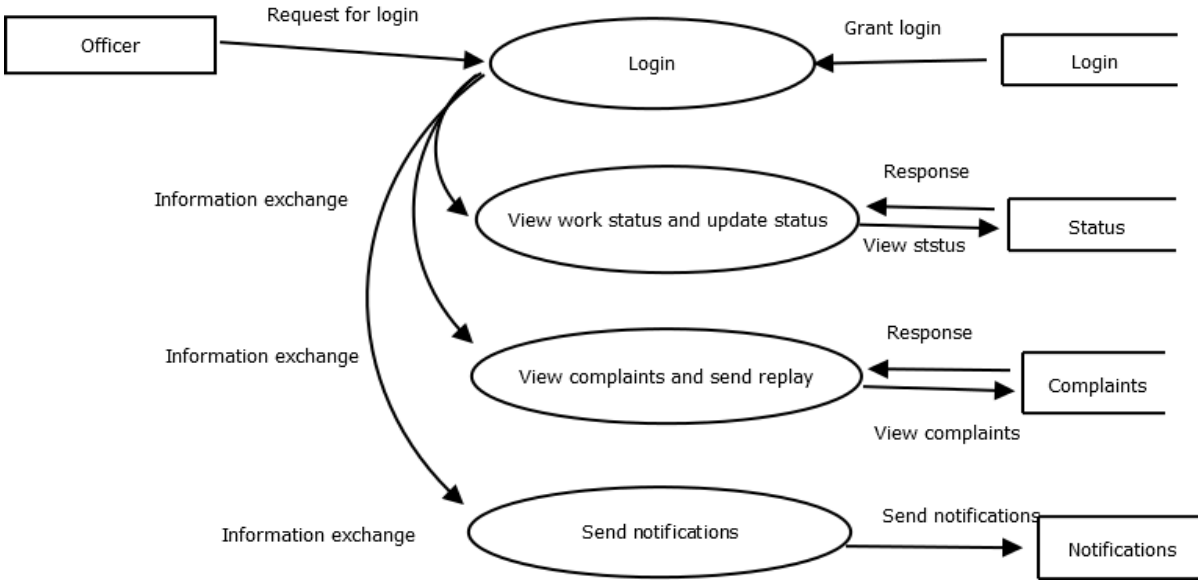

Figure A.4: Dataflow Diagram

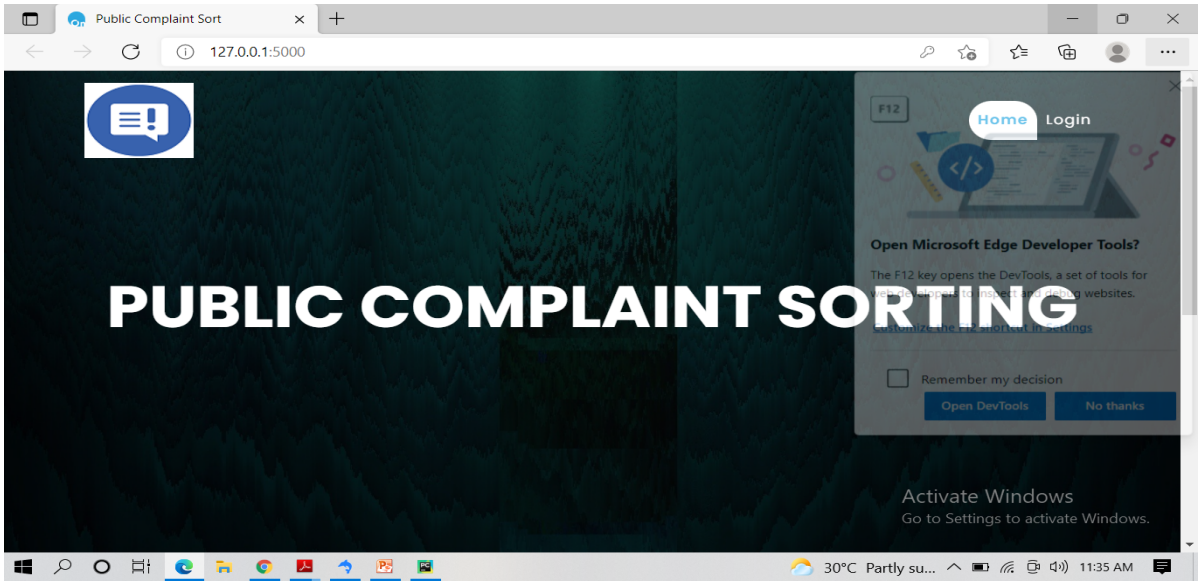

Figure A.5: User Interface

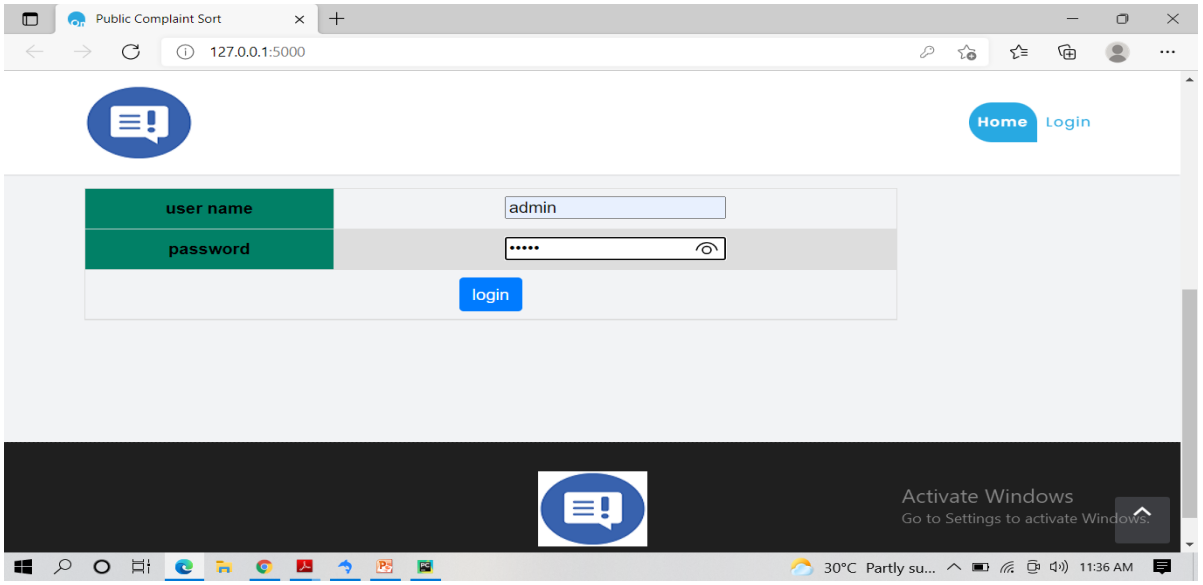

Figure A.6: user interface

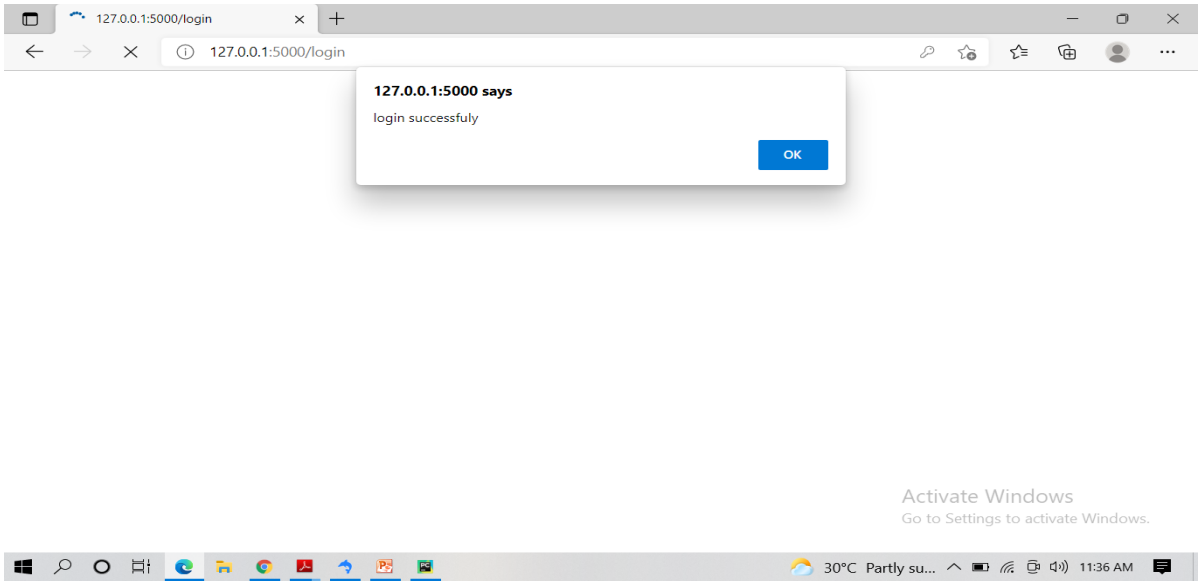

Figure A.7: user interface

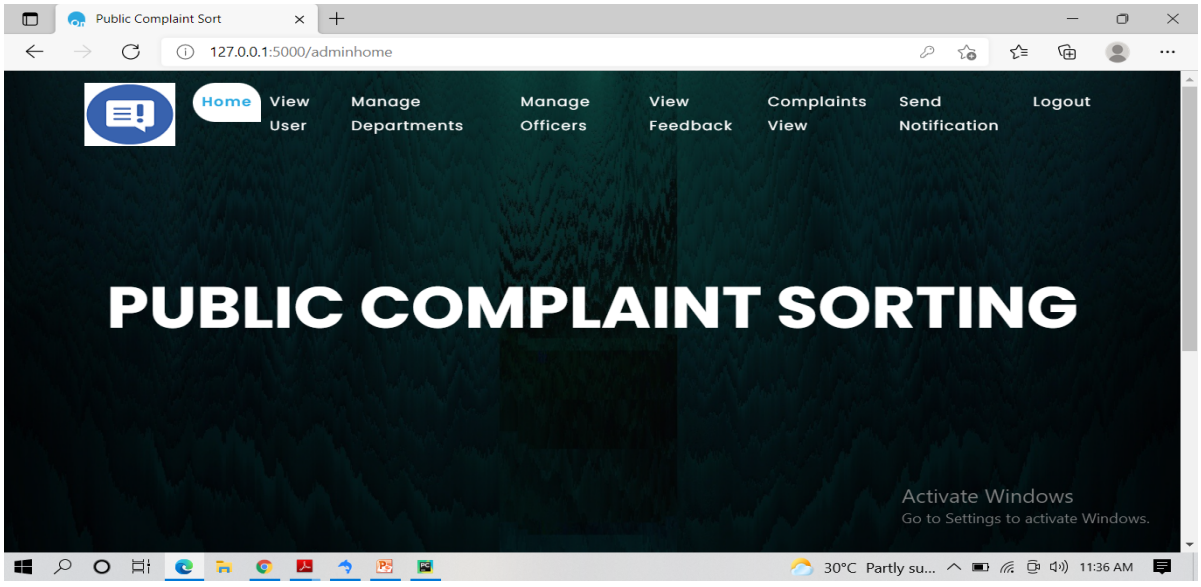

Figure A.8: user interface

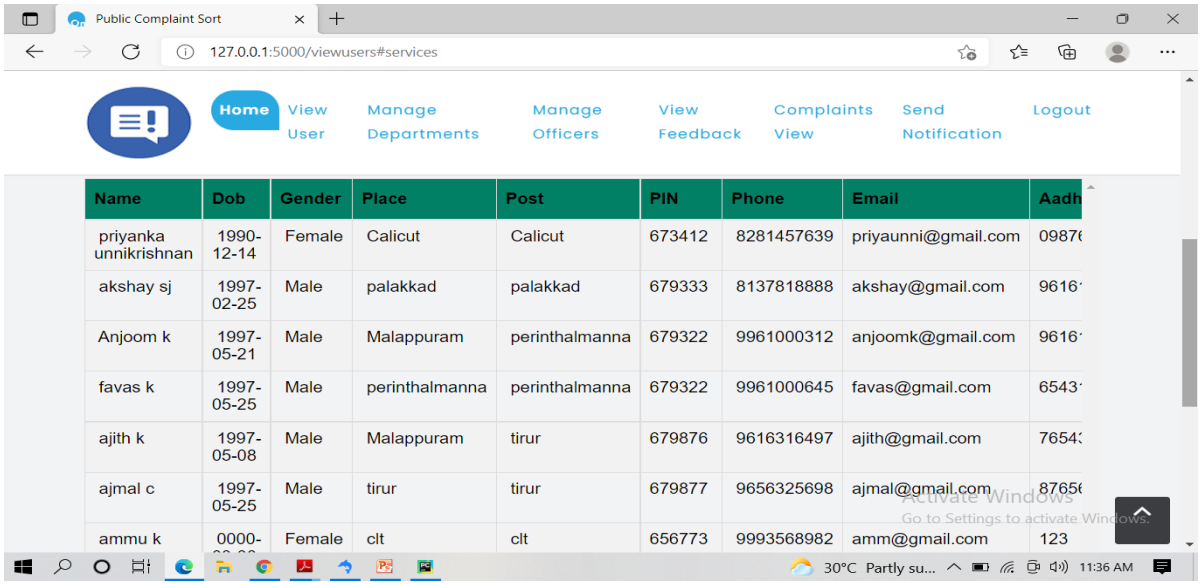

Figure A.9: user interface

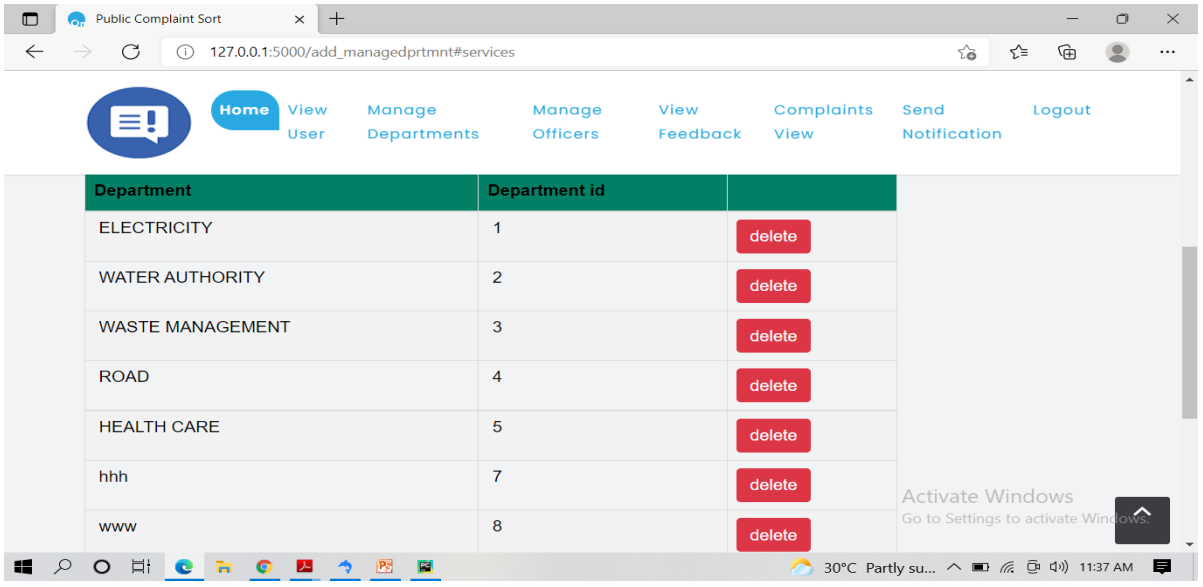

Figure A.10: user interface

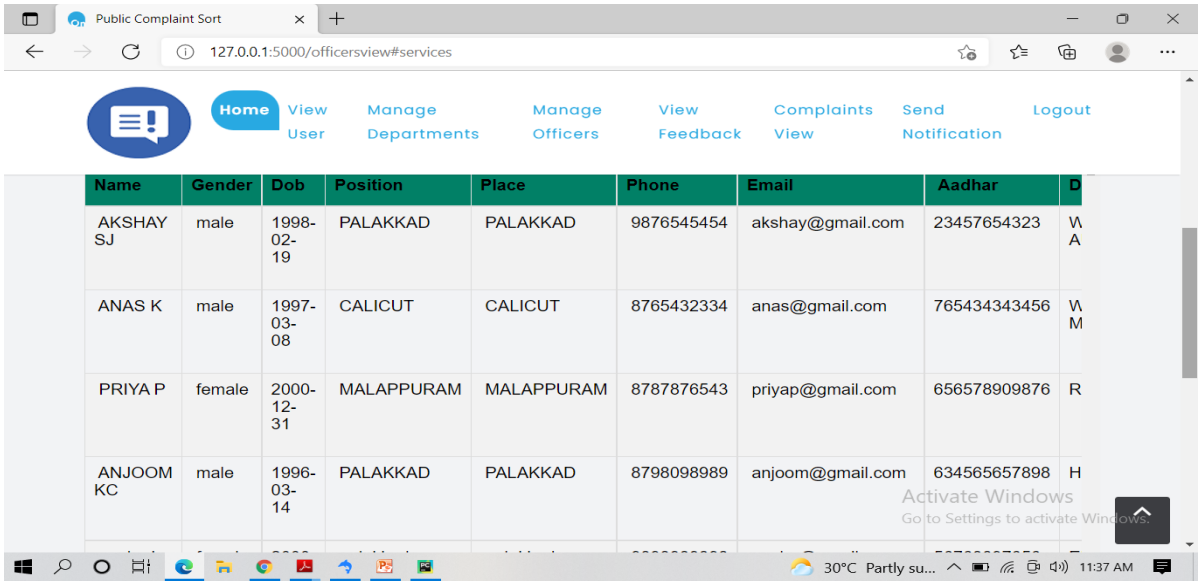

Figure A.11: user interface

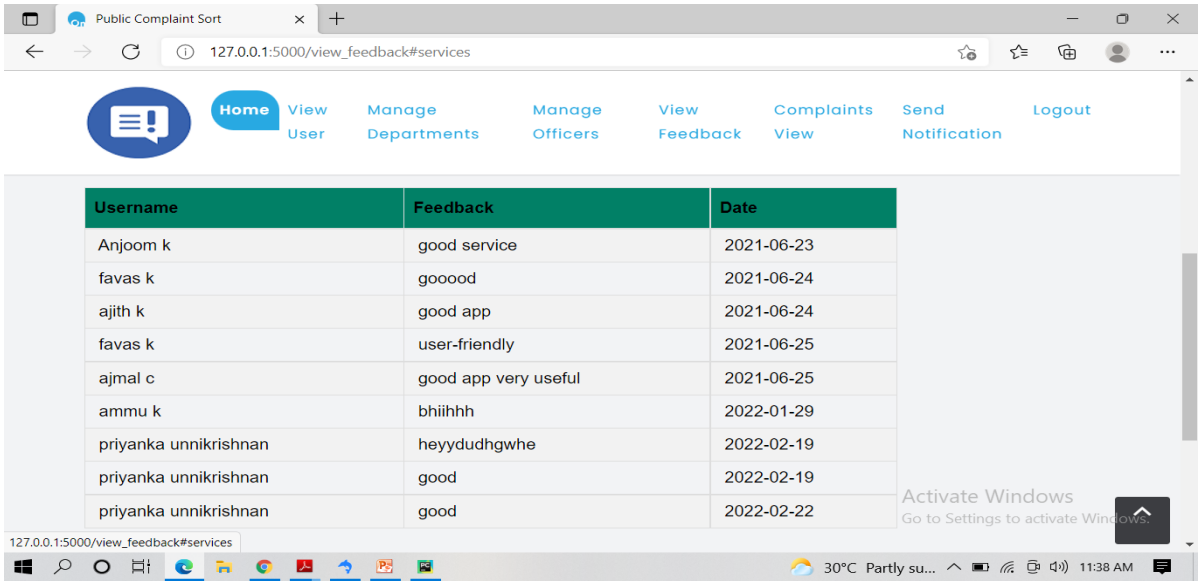

Figure A.12: user interface

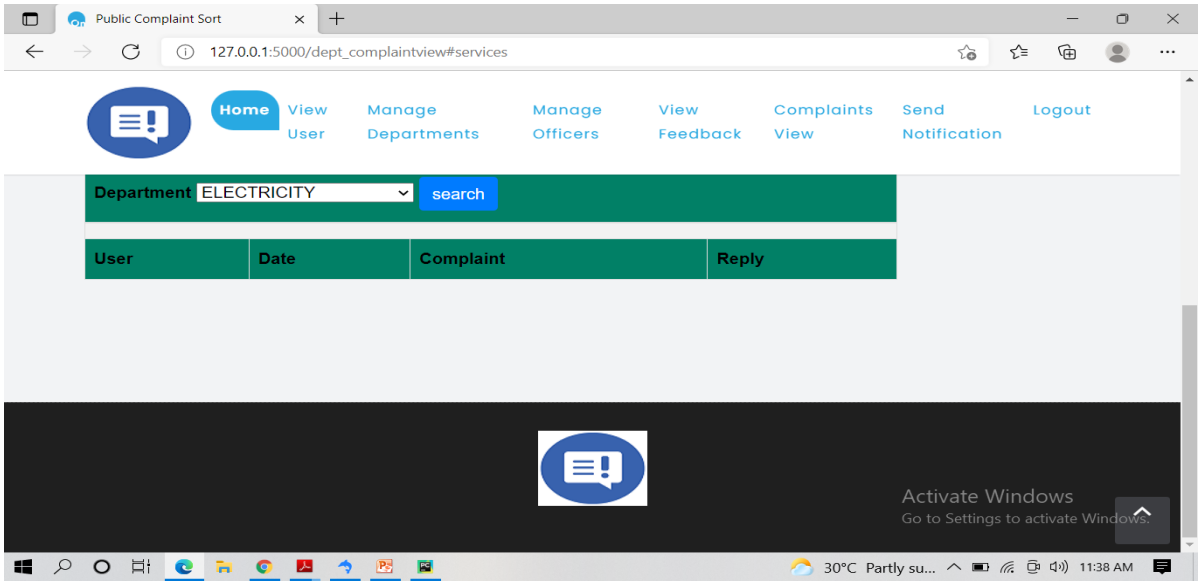

Figure A.13: user interface

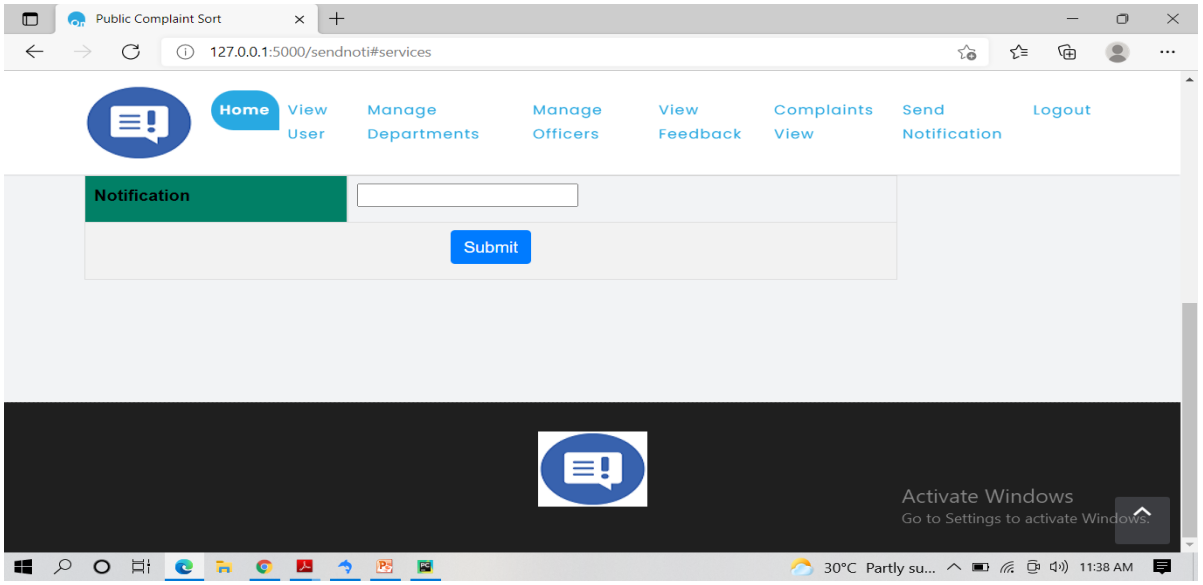

Figure A.14: user interface

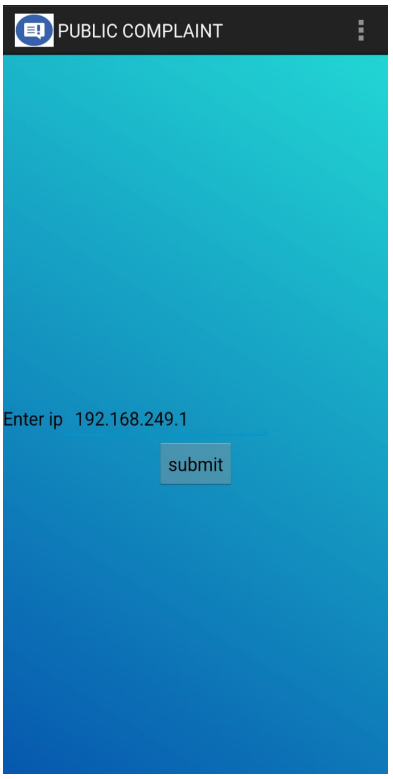

Figure A.15: user interface

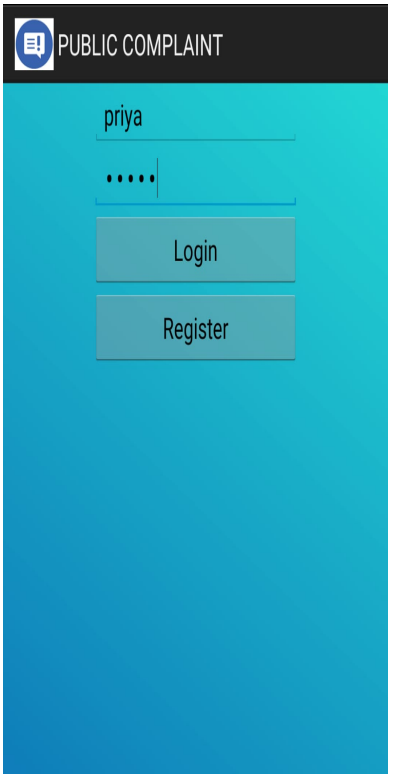

Figure A.16: user interface

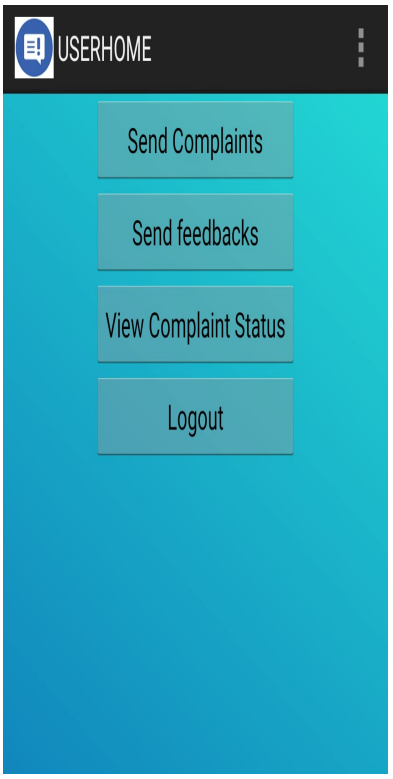

Figure A.17: user interface

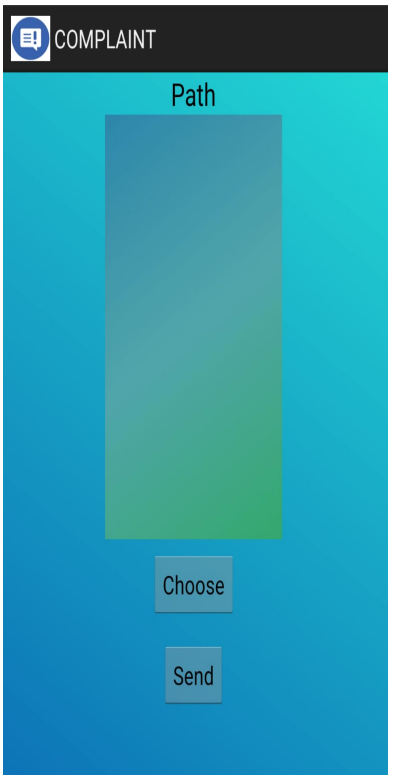

Figure A.18: user interface

Figure A.19: user interface